List Master

by Ted Birkhead and Roger Wagner

A time-saving set of Applesoft listing-related utilities
produced by
Southwestern Data Systems

ListMaster™

by Ted Birckhead and Roger Wagner

INSTRUCTION MANUAL

PRODUCED BY:

SDS™

southwestern data systems™

10761-E Woodside Avenue • Santee, California 92071
Telephone: 714/562-3670

SOUTHWESTERN DATA SYSTEMS

CUSTOMER LICENSE AGREEMENT

IMPORTANT: The Southwestern Data Systems software product that you have just received from Southwestern Data Systems, or one of its authorized dealers, is provided to you subject to the Terms and Conditions of this Software Customer License Agreement.

Should you decide that you cannot accept these Terms and Conditions, then you must return your product with all documentation and this License marked "REFUSED" within the 30 day examination period following the receipt of the product.

1. License. Southwestern Data Systems hereby grants you upon your receipt of this product, a nonexclusive license to use the enclosed Southwestern Data Systems product subject to the terms and restrictions set forth in this License Agreement.

2. Copyright. This software product, and its documentation, is copyrighted by Southwestern Data Systems. You may not copy or otherwise reproduce the product or any part of it except as expressly permitted in this License.

3. Restrictions on Use and Transfer. The original and any back-up copies of this product are intended for your personal use in connection with a single computer. You may not distribute copies of, or any part of, this product without the express written permission of Southwestern Data Systems.

DISCLAIMER

SOUTHWESTERN DATA SYSTEMS AND THE PROGRAM AUTHOR SHALL HAVE NO LIABILITY OR RESPONSIBILITY TO PURCHASER OR ANY OTHER PERSON OR ENTITY WITH RESPECT TO ANY LIABILITY, LOSS OR DAMAGE CAUSED OR ALLEGED TO BE CAUSED DIRECTLY OR INDIRECTLY BY THIS SOFTWARE, INCLUDING, BUT NOT LIMITED TO ANY INTERRUPTION OF SERVICE, LOSS OF BUSINESS OR ANTICIPATORY PROFITS OR CONSEQUENTIAL DAMAGES RESULTING FROM THE USE OR OPERATION OF THIS SOFTWARE.

** LIST MASTER **

TABLE OF CONTENTS

LISTER

COMP-LIST

LIST MASTER - INTRODUCTION

LIST MASTER is a set of Applesoft program listing-related utilities. These allow the user to make quite a number of large scale changes to any Applesoft program in a fast and efficient manner.

LIST MASTER is made up of four basic programs. APPLESPEED is a program optimizer that gives you the option of shortening variable names, combining lines, removing REMark statements, and renumbering by 1's. Most of these choices may be combined to make just the changes you want. The net result will be a program that is both more compact and which runs at optimum speed.

RENUMBER PLUS takes the basic idea of a renumber program, and takes it many steps further to provide a truly powerful utility program. In addition to being able to renumber program line numbers in the usual manner, RENUMBER PLUS allows you to preserve the logical structure to your programs. Many programmers like to put subroutines in blocks of code with their own line number ranges. For example, a basic menu routine might be written into the program starting at 5000. Other subroutines might also begin with even thousand starting lines. With other renumber programs, all this structure will be lost as the entire program is blindly renumbered, unless you're willing to manually renumber each block one at a time. With RENUMBER PLUS you can specify the beginning of logical blocks, and these will be preserved in the new listing.

LISTER provides an easy way of listing programs to your printer, along with a title and date banner. In addition, listings can be done by the page, even if your printer does not normally support this! Left, top, and bottom margins may also be set so that listings can be put in a standard 3-ring notebook. LISTER will also create standard Apple DOS textfiles of any Applesoft program. These can then be processed by various text editors like THE CORRESPONDENT, or even transmitted over phone lines by use of a modem and proper communications software like the ASCII EXPRESS II, both of which are available from Southwestern Data Systems.

Another use for a textfile form of an Applesoft program is in programs like COMP-LIST, the fourth program in the LIST MASTER set. COMP-LIST will take the files of any two programs and list out any and all differences between them. This is useful not only for detailing changes you may make with APPLESPEED and RENUMBER PLUS, but also in checking later versions of programs against earlier ones to see just what changes may have been made. COMP-LIST will also create text files of just the changes which have been made. This creates UPDATE files which will convert an earlier version to the newer one in a minimum of time. This can be handy for sending updates to other users, without having to send the actual program.

The following manual details each of these programs, and how to use them. We think you'll find them a valuable part of your program library.

->>>--APPLESPEED--->

by Ted Birkhead

Using APPLESPEED

1. When using APPLESPEED it is assumed that you may already have the program you wish to work on in memory. If this is not the case, don't worry as your program may be LOADed at any time.

2. To install APPLESPEED, place the LIST MASTER diskette (or a backup) in your disk drive and type in: EXEC AS INSTALL

3. A number of prompts will appear on the screen, APPLESPEED will load, and this banner will appear on your screen:

```
>>>>>>>> ->>>--APPLESPEED---> <<<<<<<<<<
>                                        <
>  COPYRIGHT 1981 ALL RIGHTS RESERVED    <
>                                        <
>      SOUTHWESTERN DATA SYSTEMS         <
>                                        <
>    APPLESPEED DOES THE FOLLOWING:      <
>                                        <
>  1. LISTS LINES WITH GOTO'S ETC,THAT   <
>     ARE INVALID                        <
>                                        <
>  2. DELETES REMARK STATMENTS           <
>                                        <
>  3. MINIMIZES VARIABLES                <
>                                        <
>  4. COMBINES LINES                     <
>                                        <
>  5. RENUMBERS PROGRAM BY 1'S           <
>                                        <
>  NOW ENTER THE NUMBER OF THE HIGHEST   <
>  SEQUENTIAL STEP DESIRED               <
>                                        <
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

4. Follow the instructions and enter a number from 1 to 5, corresponding to the highest sequential action you wish to accomp lish. If, for example, you only want to check your program for missing line numbers, enter a 1. To do item 1 plus deleting remark statements and minimizing variables, enter a 3.

   NOTE: The program will do all of the steps through the number that you select. You may also later decide to alter these choices. The following sections contain more information on how to do this, and on each of the specific options.

5. After entering your selection, the following message will appear:

TO RUN APPLESPEED, ENTER AN AMPERSAND (&)

6. That's all there is to it! If your program is aleady in memory entering ampersand will invoke the APPLESPEED utility. If your program is not already in memory, simply load it as you normally would, then use the ampersand.

7. To change option levels it is not necessary to re-EXEC the the AS INSTALL file. Special run time parameters can be used to modify option level, turn on or off variable shortening, and select line limits during line combining. See page 5, "Run Time Parameters" for further details.

NOTE: APPLESPEED will stay installed in your computer until you either 1) switch languages to Integer, 2) type in 'FP', or 3) run a program which resets HIMEM: itself. To remove APPLESPEED, the best way is to simply type in: FP and press RETURN. This will also erase any other program currently in memory.

Available Options

When the menu is presented, you have a number of choices about the extent to which you wish to change your program. When you select a given option level, that function and all levels below it will be executed unless suppressed. Option suppression is described in the section immediately after this one, called "Run Time Parameters".

1. LIST MISSING LINE NUMBERS:

   All references that transfer control to another line number (such as GOTO's, GOSUB's, etc.) will cause the APPLESPEED routine to check to insure that the referenced line number really exists. If it does not, the number of the line and the offending reference are listed in the following format:

LINE #	INVALID #
150	225

   This would indicate that line #150 contained a reference to line #225 (such as a GOTO 225), when 225 does not actually exist in the compressed listing.

   This output can be directed to a printer by selecting it as an output device before initiating a run. This is usually done by just typing in PR#1 or equivalent, just before entering the ampersand 'RUN' command.

   Control S will temporarily halt the output, just as it does with the APPLESOFT LIST command.

If any errors are noted, the program will abort. You should fix invalid references and rerun.

2. DELETE REMARK STATEMENTS:

   REMark statements are removed and any references that transfer control to a line that does not now exist are redirected to the next line.

   REMarks take up one byte for each character in the Remark. They also slow down execution time as the computer skips over them.

3. REDUCE VARIABLES:

   All variables that are more than two characters long are reduced to two characters.

   Shorter variable names are more quickly accessed by the Apple, thus speeding up your program.

4. COMBINE LINES:

   All lines that can legally be combined are connected by a colon (:). Example:

   ```
   10 HOME
   20 PRINT "THIS IS A TEST."
   ```

   would become:

   ```
   10 HOME: PRINT "THIS IS A TEST."
   ```

   This would save not only 5 bytes for each leading line number removed, but increase the operating speed since the Apple no longer has to deal with skipping over the old line #20.

5. RENUMBER LINES BY 1'S:

   Lines are renumbered with an initial number of one and an increment of one. All references such as GOTO's, GOSUB's, etc. are also renumbered as is appropriate.

   In Applesoft, line numbers use 1 byte for each digit of the number. Thus 'GOTO 1' takes only one byte for the number vs. 5 bytes for 'GOTO 30000'. Multiplied by the number of times 'GOTO's, etc. appear in most programs, this usually means big savings in program size.

Comments:
---

1. A tremendous amount of work is done during program execution. Give APPLESPEED time to finish the job, and don't press the reset key. If you do, the program being compressed will be destroyed.

Experience will show that a large program containing complicated control transfers can require a couple of minutes for each step in reducing variables, combining lines, and renumbering.

2. Please do not run APPLESPEED against your only copy of a program. The name of the game is backup, and 'gremlins' can make a lot of extra work for those that don't follow good backup procedures.

3. If you want to change your program after a compression run it is best to only go through step three. You will find that a program with the lines combined can be secure even to its author.

4. APPLESPEED tables are built back from HIMEM. If your Applesoft program is about to be overlayed by any table, APPLESPEED will abort with the error message: "ABORT PROGRAM TOO LARGE".

Run Time Parameters

1. Provided that HIMEM: is not changed by a DOS boot, POKES, FP, etc, APPLESPEED will remain intact. Each time an "&" is entered, it will continue to operate at the option level selected at load time. However, entry of special parameters after the "&" will allow for:

   A. Resetting the option level.

   B. Suppression of the variable shortening even though an option level was selected that included it.

   C. Selection of limits for combining lines. (Zero parameters will supress all combining)

2. Here are the rules.

   A. Each parameter must begin with a special command letter telling the system which option to modify, and be followed by a number, or numbers indicating the modification.

LETTER	OPTIONS & NUMBER(S)
L	Level of highest option desired, (1-5).
V	Variable shortening, (0=off, 1=on).
C	Combining lines, (First number=start-at line, second number=stop-before line).

   B. A comma must be used to separate parameter sets and numbers within parameters.

   C. A colon must be entered to indicate the end of parameter input.

D. Parameters may be entered in any order, but no spaces are permitted.

E. Examples:

1. &L4,V0,C100,1000:

Where: &=Call APPLESPEED; L4=Set level of highest option to 4,(Combine lines); V0=suppress variable shortening function; C100,1000=Start combining with line 100, stop before line 1000; :=End of parameter input.

2. &L5,C0,0:

Where: &=Call APPLESPEED; L5=Set level of highest option to 5, (All options); C0,0=Start combining lines with line 0, stop combining lines before line 0, (Effect will be to null joining lines); :=End of parameter input. In this case variables will be shortened, since not suppressed with a V0.

3. &L4:

Where: &=Call APPLESPEED; L4=Set level of highest option to 4, (Do everything except renumber the program by 1), :=end of parameter input. This is the simplest form of parameter input.

NOTE: Since the "variable shorten flag" was not set to "off", (0), in example 1, it will continue to be off in examples 2 and 3. So, once the variable shorten flag is set, it is not necessary to restate its value unless you wish to toggle it.

+++++ RENUMBER PLUS +++++

by Ted Birkhead

Using RENUMBER PLUS

1. Using RENUMBER PLUS is very similar to APPLESPEED. The program you wish to renumber may be safely left in memory when installing RENUMBER PLUS. If you do not have a program loaded when doing the installation, any program may be loaded afterwards without having to re-run the installation program.

2. To install RENUMBER PLUS, place the LIST MASTER diskette (or a backup) in your disk drive and type in: EXEC RENUMBER PLUS

3. A number of prompts (]) will appear on th screen, after which a help list will appear.

   NOTE: This list is provided solely as an aid in remember-in the commands, and no particular response is required.

4. RENUMBER PLUS is now loaded. Next, if the program you wish to renumber is not already in memory, simply LOAD it as you would do normally. Then follow instructions A-D to renumber the program using whatever parameters you desire.

   A. Enter an "&" followed by a carriage return. if you have a program in memory, it will be renumbered with the value of the first line number equal to 10, and each succeeding line number incremented by 10. All internal transfers of control will be updated to the new corresponding line numbers.

   B. Prior to renumbering, tests will be made to ensure that a program exists. If it does, it will be checked to ensure that there are no GOTO's etc without corresponding line numbers. If there are, the program will abort.

   C. A series of execution prompts will appear telling what RENUMBER PLUS is doing. Each one of these is explained in the section of the manual titled EXECUTION PROMPTING. The last prompt to appear will be "*RENUMBERED*", meaning that the job is finished.

   D. This is what the entire series of events should look like:

```
EXEC RENUMBER PLUS
RENUMBER PLUS READY
&
TESTING FOR NUMBER OVERLAP & OVERFLOW
TESTING FOR INVALID CONTROL XFERS
RENUMBERING
ADJUSTING LINE # REFS
*RENUMBERED*
```

PROGRAM OPERATION USING OPTIONS

Options are triggered by key letters. With the exception of the ampersand, which must appear first and in column one, they may appear in any order, with the proper format of course.

Spaces are not allowed in the option string. All options must be separated by a comma, and the last one must be followed by a colon.

OPTION TABLE

Key Letter	Allowable Values	Command
B	0-63,999	Begin renumbering at this line #.
E	1-63,999	End renumbering prior to line #.
V	0-63,999	Value of 1st # to insert.
I	1-63,999	Increment succeeding lines by #.
P	2-4	Preserve line #'s ending in this # of zeroes or more.
O	none	Overide the check for invalid control transfers.
X	none	Extend the maximum values of all options except P, to 65,535.
S	none	Store the current program.
M	none	Merge the stored program into the current program.
R	0-63999	Re-store the saved area contents by inserting it into the current program at the line number following the r parameter.

EXAMPLES OF RENUMBERING OPTION STRINGS

STRING	RESULT
&B100,E1000,V200,I500:	Renumber program starting at line # 100, (B100) end renumbering before line # 1000, (E1000) use an initial value of 200, (V200) & increment suceeding line by 500, (I500).
&P3,X,O:	Renumber entire program by defaults, however each time a line is found ending in 3 or more zeroes, do not change the number but use it as the initial number for continuing the renumbering (P3). If necessary let the line numbers exceed the maximum of 63,999 (X). Overide the test for invalid control transfers (O).

STORING YOUR PROGRAMS IN MEMORY

&S Entry of "&S" will store off a copy of the current program. If Begin & End options are specified, saving will be of the line numbers specified. Saving a program or section of one is prerequisite to Merging or Re-storing, since something has to be in storage to Merge or Re-store into the current program. Examples follow:

STRING	RESULT
&S:	The entire current program is stored off to high memory.
&S,B200,E1000:	The section of the program starting at line 200, and ending at (not including 1000), will be stored.

MERGING PROGRAMS

&M Entry of "&M:" will merge any stored program into the current program. Other options entered with the same string will be ignored. If any line #'s match, the merge will abort prior to any merge execution.

RETRIEVING STORED PROGRAMS

&R The saved program will be renumbered with the first # equal to the value following the R. The current program will be renumbered so that the saved area will merge into it at the specified point. Since the program or subroutine in the saved area may not relate to the current program, be sure to check the control transfers in the section merged into the current program. Transfers to lines within itself will be correct. Transfers outside will probably not be correct. An example of the Re-store option follows.

STRING	RESULT
&R200:	The lines in the saved area will be renumbered starting with 200. The current program will then be renumbered starting at 200 with the value of the first renumbered line equal to the last line number in the renumbered save area plus the increment. This will be followed by a merge of the save area into the current program. All of this is done automatically!

Before attempting to use RENUMBER PLUS, please study the following sequences of program modifications. The sequences were produced by loading small example programs, listing them and then entering and executing RENUMBER PLUS paramater strings. Each resulting program was then listed and remodified as required.

Normally program prompts, telling what is happening will appear after the execution of a parameter string. These have been omitted in the following examples in the interest of saving space.

```
LOAD EX1

LIST

5 GOTO 10 : REM EX1
10 GOTO 15 : REM EX1
15 GOTO 20 : REM EX1
20 GOTO 5 : REM EX1

&  (Renumber using all of the defaults.)

LIST

10 GOTO 20 : REM EX1
20 GOTO 30 : REM EX1
30 GOTO 40 : REM EX1
40 GOTO 10 : REM EX1

&V100,I100:  (Renumber the entire program, the default, as no B or
              E parameters were given. Let the value of the first
              number entered be 100 (V100), increment each suc-
              ceding line number by 100 (I100).

LIST

100 GOTO 200 : REM EX1
200 GOTO 300 : REM EX1
300 GOTO 400 : REM EX1
400 GOTO 100 : REM EX1

&B200,E400:    (Renumber the program beginning at line number 200.
            Since there is no V parameter for the value of the
            first number to be inserted, use the default of the
            current number of the line. Let the increment be the
            default of 10, and end the renumbering before line
            number 400 (E400).

LIST

100 GOTO 200 : REM EX1
200 GOTO 210 : REM EX1
210 GOTO 400 : REM EX1
400 GOTO 100 : REM EX1
```

```
LOAD EX2

LIST

11 GOTO 21 : REM EX2
21 GOTO 2000 : REM EX2
2000 GOTO 3234 : REM EX2
3234 GOTO 11 : REM EX2

&P2:        (Renumber the entire program using all of the defaults,
         however,  whenever  a line is found that has a number end-
         ing  in 2 or more zeroes (P2), Preserve the number of that
         line,  and  use  it as the beginning number for continuing
         the renumbering.

LIST

10 GOTO 20 : REM EX2
20 GOTO 2000 : REM EX2
2000 GOTO 2010 : REM EX2
2010 GOTO 10 : REM EX2

&S:         (Store a copy of the current program into high memory.)


LOAD EX3

LIST

5 GOTO 25 : REM EX3
25 GOTO 90 : REM EX3
75 GOTO 5 : REM EX3

&M:           (Merge  the  stored program into the current program)

LIST

5 GOTO 25 : REM EX3
10 GOTO 20 : REM EX2
20 GOTO 2000 : REM EX2
25 GOTO 75 : REM EX3
75 GOTO 5 : REM EX3
2000 GOTO 2010 : REM EX2
2010 GOTO 10 : REM EX2

LOAD EX1

LIST

5 GOTO 10 : EX1
10 GOTO 15 : EX1
15 GOTO 20 : EX1
20 GOTO 5 : EX1

&S:         (Store a copy of the current program into high memory.)
```

```
LOAD EX2

LIST

11 GOTO 21 :REM EX2
21 GOTO 2000 :REM EX2
2000 GOTO 3234:REM EX2
3234 GOTO 11 :REM EX2
```

&R500: (Re-merge the saved area into the current program at line number 500.)

```
LIST

11 GOTO 21 : REM EX2
21 GOTO 540 : REM EX2
500 GOTO 510 : REM EX1
510 GOTO 520 : REM EX1
520 GOTO 530 : REM EX1
530 GOTO 500 : REM EX1
540 GOTO 550 : REM EX2
550 GOTO 11 : REM EX2
```

Note in this example that both of the involved areas, save and current program have had the internal references adjusted. The save area was considered as a separate program, with internal references adjusted only to numbers existing within itself. The current program is "opened up" to make room for the save area. References are again adjusted within its own content before the save area is entered.

```
LOAD EX4

LIST

5 GOTO 200
50 GOTO 60
60 GOTO 50
```

&O: (Renumber the entire program using defaults, but Overide the check for invalid control transfers)

```
LIST

10 GOTO 200
20 GOTO 30
30 GOTO 20
```

PROGRAM PROGRESS REPORTS

RENUMBER PLUS executes in a series of distinct phases. to keep you posted on its progress it outputs a series of prompts. A list of these prompts and the meaning of each follows:

	PROMPT	MEANING
1.	PARSING PARAMETERS	Attempting to decipher the "&" input string and parse into its components.
2.	TESTING PARAM RELATIONSHIPS	Phase 1 was sucessfully completed. Looking for parameter conflicts & error conditions.
3.	TESTING FOR NUMBER OVERLAP & OVERFLOW	No fatal error in phase 2. Now doing a trial renumber without disturbing the current program. Testing for overflow and violation of number integrety.
4.	TESTING FOR INVALID CONTROL TRANSFERS	So far all is ok. Now looking for control transfers to lines that don't exist. If any are found, they are listed and the program aborts.
5.	RENUMBERING	If this prompt appears, there were no invalid control transfers & renumbering has begun.
6.	ADJUSTING LINE # REFS	When this prompt appears, renumbering is done. The system is now adjusting all line references that are objects of control transfers to conform to the new line #'s.
7.	*RENUMBERED*	The renumber sequence is completed.
8.	*STORED*	A copy of the current program has been stored in high memory.
9.	*MERGED*	The stored area has been merged into the current program.
10.	*RESTORED*	The stored area and the current program have been renumbered, and the merge has been executed.

ERROR MESSAGES

RENUMBER PLUS outputs the following error messages:

	MESSAGE	EXPLANATION & POSSIBLE CURE
1.	VALUE > 63,999 REQUIRES X OPTION	Maximum exceeded. If caused by a qualifier value, either use X option or reduce value. If caused during renumbering try use of X option & or a smaller Increment.
2.	I = 0 IS INVALID	Increment must not be zero. If it is all line numbers will be renumbered to the same number.
3.	P OPTION MUST BE 2, OR 4	Change to one of the permissible values.
4.	BEGIN POINTS TO PROGRAM END	No renumbering can take place. Readjust Begin parameter.
5.	B MUST BE < V, IF B NOT 1ST LINE	Overlap. Make V larger than the number of the B option.
6.	OVERLAYED PROGRAM CLEARED	The copy of the current program has overlayed the original. The damaged original is cleared.
7.	16 BITS EXCEEDED	Overflow. Try a smaller increment.
8.	63,999 EXCEEDED AND NO X OPTION	Use X option & or a smaller Increment option.
9.	LAST CHANGED NOT < NEXT	Overlap. Try a smaller Increment.
10.	P # < OR = PREVIOUS LINE#	Overlap. Try a smaller Increment.
11.	NO PROGRAM IN MEMORY	Load the program you want to renumber, or merge into.
12.	IN LINE ---- INVALID # ----	Invalid control transfer. Fix or use the override option.
13.	DUP ERR	A matching line number was found during the merge tests. Renumber either the stored program, or the current program to resolve the conflict.

14. TABLE WOULD OVERLAY	Because of the extreme length of the program it would be overlayed by required tables. Shorten it if possible. If the stored area contains a program, get rid of it with a merge or restore.
15. SAVE IN USE	Only one program at a time can be stored. Clear the save with a merge or a re-store.
16. BEGIN IS NOT < END	Impossible condition. No renumbering would occur.
17. EMPTY SAVE	Something was needed to merge etc. Put it in storage.
18. PROG EXP ERR	The program has been renumbered. However, during line adjustment expansion was about to overlay protected memory. The renumbered program is alive and well, however one or more line # references have not been changed.
19. MERGE EXP ERR	Unlikely you will see this, but it means that the merged line was about to overlay the saved area. Try merging in two parts.
20. SAVE EXP ERR	Same as 17 however the save area has been deleted, since the user cannot access it.
21. LINE MAX ERR	The maximum allowable line size of 238 characters was about to be exceeded by the insertion of a larger #. It and any following numbers in the line were not replaced.
22. LINE TOO LONG	Applesoft has a maximum line size of 239. One or more lines exceed that value. Change the lines.
23. INVALID OPTION OR FORMAT	Correct the option or format.

** PROGRAM LISTINGS: LISTER **

LISTER will list all or part of an Applesoft program to either a printer or a disk textfile. To use it, load the program you want listed (if it is not already in memory), and then type in ´EXEC LISTER´. (LSTR.PROG will be the actual program loaded and run by the LISTER file).

After pressing any key to clear the title banner, this menu will be displayed:

```
PROGRAM LISTING ROUTINE

PROGRAM NAME?

TODAY´S DATE?

PLEASE INDICATE WHICH FUNCTION:

  (1) LISTING
  (2) TEXTFILE TO DISK

(DEFAULT=LISTING)

WHICH?
```

The first two questions asked will be for the program name and the date. Both of these may be defaulted on by pressing RETURN alone. You have at this point a choice between two functions:

1) LISTING:

This will list a line range of your choice to your printer, enclosing the program name and date in an attractive banner.

(There are a number of format options which may be used in the listing. See the following section on printer modifications for more information on this.)

(INVOKING LISTER WITH ´&´)

When both this and the next section are done, you will have the choice of another listing or to return to your program. If you choose to return, you can optionally enter either ´/´ or ´2*´. If you do, LISTER will be put on ´hold´ so that you can recall it later with the ´&´ command.

2) TEXTFILE TO DISK:

This will create a text file out of the line range given. The name of the file created will be whatever name you give as the ´program name´ in the initial question. It is therefore recommended that you always use either a different name when saving textfiles, or add a suffix like ´.FILE´ to the program name when entering it. This will avoid conflict between the textfile and the original Applesoft program during the save.

The new file may then be used by some of the various text editors on the market (including our own ´CORRESPONDENT´ with which this documentation was produced), or EXECed into other Applesoft programs as you wish. As in (1) you will be given the repeat/return to program option when done.

Pgs. 76,77 of the Apple DOS 3.2 Reference Manual discuss some of the aspects of programs converted into text files.

One use for the textfile form of programs is for easy compiling of subroutines into a larger program. The various modules may be converted to text files, and then saved as a program library for later use as you write new programs.

In this kind of usage, it is also a good idea to standardize the line numbering of the various kinds of routines. For example you might decide to always put menu routines at line 5000, DATA statements at 20000, error handling routines at 10000, etc.

With this technique, you can always safely merge your routines into another program without fear of overwriting a routine that might already be there.

For example, try this practice exercise:

1) Type in ´NEW´, and then this program (Applesoft):

```
5000 REM MENU ROUTINE
5010 HOME
5020 PRINT: PRINT: PRINT "1) END PROGRAM"
5030 PRINT: INPUT "YOUR CHOICE:";I$
5040 END
```

2) SAVE this program as a backup under the name MENU 1

3) Convert it to a textfile by putting the LIST MASTER diskette in the drive and typing in: EXEC LISTER

4) Put the diskette with MENU 1 back in the drive, and proceed with creating a textfile, as described earlier. Name the textfile, MENU.T

5) Put LISTER on ´hold´ (command=´/´), type in ´NEW´ and enter this program:

```
10000 REM ERROR-HANDLING ROUTINE
10010  IF  PEEK  (222) = 255 THEN PRINT "CONTROL-C":RESUME
10020 POKE 216,0:END
```

6) SAVE as a backup, under the name ERR 1, then recall LISTER by typing in an ampersand (&). Repeat the text file creation process, this time naming the file ERR.T, and also saving it on the same diskette as MENU.T.

7) Now for the final excercise. Type in ´NEW´ to clear memory, and LIST to confirm the absence of any program.

8) Now type in: EXEC MENU.T

A series of prompts should appear. When they are done, type in LIST. The program routine for the menu should have been automatically entered. Now type in: EXEC ERR.T

Typing LIST now should reveal the addition of the error-handling routine.

Whenever you ´EXEC´ a file, it is the same as if you had typed in its contents by hand. In the case of the program files, it is just as if you had typed in the actual program listings contained in the files.

You may also wish to load an expendable HELLO program or equivalent, and verify to yourself that these routines will also EXEC into programs already started.

LISTER - PRINTER MODIFICATIONS

The listing itself will be automatically formatted if you desire. You can set the page length along with top and bottom margins (to skip the page perforation), and you may also set the left margin, to allow listings to be put in notebooks.

If you do not want the formatting at all, simply retype the line changing FF = 1 to FF = 0. Then save LSTR.PROG back to the diskette and skip the remainder of this section.

To set these parameters, load the program LSTR.PROG and examine line #110.

```
]LIST 110

110 PW = 80:PS = 1:PL = 66:MB = 3:
    MT = 3:LM = 7:KF = 0:FF = 1:SF=0
```

The variables are as follows:

PW: Your printer width. Can be set from 40 to 132.

PS: Printer slot. Usual value is 1.

PL: Page length. The number of printed lines per each sheet of printer paper. The usual is 66, although this can vary.

MB: Margin (Bottom). Number of blank lines at the bottom of every page.

MT: Margin (Top). Number of blank lines to put at the top of each page after the first page.

MT and MB work together to form the perforation skip for a listing. If your printer does an automatic perf skip, then set PL equal to your page length minus the number of lines supplied by your printer during the skip. Then set MT and MB to 0. For example, if you had an IDS 460 (Paper Tiger) that always skipped 6 lines at the perforation, you would set up line 110 like this:

```
110 PW = 80:PS = 1:PL = 60:MB = 0:
    MT = 0:LM = 7:KF = 0:FF = 1:SF=0
```

LM: Left Margin. The number of blank spaces to the left of each printed line. This provides a left margin so that listings may be put in notebooks.

KF: Key Press Flag. When set to 1, the listing will stop at the bottom of each page to allow you to insert a new sheet of paper. Pressing any key on the keyboard will resume the listing at that point. Set PF = 0 for continuous printer paper.

FF: Format Flag. If you do not want this output routine to be used at all, set FF = 0. FF = 1 means the parameters discussed here will be used in all listings.

SF: Serial Flag. Set this depending on what kind of printer interface card you have. If you don't have a printer, leave SF=0.

Line number 260 actually turns on the printer, and you may wish to modify this line to agree with what you normally do to turn on the printer in your own programs. For instance, serial printers usually do not require the 'CHR$(9);PW;"N";' part of the sequence.

The printer is turned off on line number 540. The CHR$(12) on line 540 is a top-of-form command which on some printers will automatically advance the paper to the top of the next page. Even if your printer does not normally recognize this, with LISTER, the your printer will temporarily have this feature. If you wish to disable it, simply remove the PRINT CHR$(12) from line 540.

** COMP-LIST **

From time to time you may wish to compare two Applesoft program listings for any differences that may exist. COMP-LIST will allow you to do this easily, with the option of either a printed list of the differences, or a disk text file, which can then be used to create an 'update file' which when EXECed, will convert one version of a program to another.

One of the main reasons for using such an EXEC file is in cases where another user may have made their own modifications to a program, and you wish to give them an update without forcing them to rewrite all their own changes. By sending an EXEC file, you can update just those portions which have changed in your own version of the program.

Using COMP-LIST is very straight forward. First use LISTER to create a text file out of each of the programs you wish to compare. Having done that, just place the LIST MASTER diskette in the drive and type in: RUN COMP-LIST

A brief set of instructions will appear, and the following questions asked:

1) CREATE AN UPDATE FILE? (Y/N) (DEF=N)

If you wish to create the update file as described earlier, enter 'Y' and press RETURN, otherwise, press RETURN alone. A 'Y' response will create the update file under the obscure name 'UPDATE'. You may wish to rename this later to reflect its actual contents.

The next question is:

2) PRINTER ON? (Y/N) (DEF=N)

If you would like to have a printed copy of the file differences, respond 'Y' here. Otherwise press RETURN alone. If your printer does not seem to work properly here, see the section on Printer Modifications for COMP-LIST, later in this documentation.

Next it will display:

3) SELECT FILE #1
(PRESS ANY KEY FOR CATALOG....)

As you press a key the CATALOG should appear. Enter the name of the text file to use as the "older" program. Do not enter the name of an Applesoft program here, only that of the textfile which you created using LISTER.

If the file you want is not on the current diskette, you may CATALOG again by pressing RETURN alone.

After entering the first name, the diskette will CATALOG again, and ask for the 2nd file name. This will be treated as the "newer" file.

```
4) SELECT FILE #2
(ANY KEY FOR CATALOG...)
```

Please note that both files must be "online" at the same time to be compared. This means that they must be on the same diskette.

If you do name a file not on the diskette, or a non-text file, you will get the error message: IMPROPER FILE. ANY KEY TO TRY AGAIN. In such a key the file name question would then be repeated.

As the files are compared, any differences found will be listed to screen or printer, as selected. (Note: When printer width is set greater than 40 columns, as described in the section on Printer Modifications, screen display AND printer output CANNOT occur simultaneously.)

Differences found will be listed in terms of lines added, deleted, or changed. Consider these sample programs:

```
10 REM TEST PROGRAM #1
20 HOME
25 PRINT "PROGRAM 1"
30 END
```

```
10 REM TEST PROGRAM #2
15 PRINT "PROGRAM 2"
20 HOME
30 END
```

Program #1 will be treated as the "older" program, #2 as the "newer" one. When textfiles are made of these two programs, and COMP-LIST used, the following list would be generated:

```
     OLD LINE:
 10 REM TEST PROGRAM #1
     CHANGED TO:
 10 REM TEST PROGRAM #2

     LINE ADDED:
 15 PRINT "PROGRAM 1"

     LINE DELETED:
 25 PRINT "PROGRAM 2"
```

At any time during the listing, the comparison may be temporarily halted by pressing any key. The process will resume by pressing any key again. Pressing the ´ESC´ key will terminate the comparison.

COMP-LIST: PRINTER MODIFICATIONS

As with LISTER, the printer output can be automatically formatted if you desire. The only difference between LISTER and COMP-LIST is that: 1) the program to modify will be ´COMP-LIST´ and 2) the line with the printer values on it will be #61900.

Otherwise, see the section immediately preceeding on LISTER PRINTER MODIFICATIONS for a desription of the possible variables and their appropriate values.

If problems occur, you may wish to change line #80 of the program ´COMP-LIST´ to appear as the ´PRINTER TURN-ON LINE´ does in your own programs. You may alter this line in any way necessary to obtain proper initialization of the printer.

MAKING BACK-UP COPIES OF LIST MASTER

To make back-up copies of the LIST MASTER diskette, you should use the copy program provided on the original diskette. To run this, simply boot on the diskette, and as the drive light comes on, press the 'C' key on the keyboard. A banner will appear and pressing any key will proceed with the copy program. You can make up to 3 copies of the original diskette.

NOTE: THE WRITE-PROTECT NOTCH ON BOTH THE ORIGINAL AND COPY DISKETTES MUST BE UNOBSTRUCTED WHEN MAKING THE COPY.

Because DOS 3.3 is much more media sensitive, only the best quality diskettes should be used for your back-ups.

If you should accidentally delete a file from one diskette, you can LOAD the file from your master or one of your copies and SAVE it back on the disk from which it was deleted.

It is recommended that you make a back-up immediately upon purchase, place the original in a safe place, and then use the copy for your daily work. Make any alterations necessary to the copy only. This will assure that you will always have a reliable version of LIST MASTER on hand.

Some individuals like to make general utility disks with all of their utilities on one diskette. This can still be done with LIST MASTER. The only constraint is that you must start with the back-up copy of LIST MASTER as the 'core' diskette and place the other utilities on it. Feel free to replace the HELLO program (on the back-up ONLY) with any other program of your choice. However, DO NOT use utilities that will alter the DOS or overall format of the disk.